Estimating the Efficiency of Information Technology for Domain-Specific Mathematical Modelling

CONFERENCE PAPER · APRIL 2015

Vitaliy Mezhuyev
Berdyansk State Pedagogical University

Estimating the Efficiency of Information Technology for Domain-Specific Mathematical Modelling

Vitaliy Mezhuyev, Mazlina Abdul Majid
Faculty of Computer Systems and Software Engineering
University Malaysia Pahang, Gambang, Malaysia
mejuev@ukr.net, mazlina@ump.edu.my

Ravi Samikannu
Faculty of Electrical Engineering,
Selvam college of technology, India
drravieee@gmail.com

***Abstract*—The paper analyses the efficiency of the Information Technology (IT) for Domain-Specific Mathematical Modelling (DSMM). IT DSMM was developed to meet the shortcomings of the IT for Domain-Specific Modelling (DSM). IT DSMM allows us to develop metamodels in the different mathematical semantics and thus to increase the level of adequateness of models to the modelled domains. It also reduces an amount of operations needed for the development of metamodels and models of software systems. The optimization problem of choosing the most effective IT by selection and comparison of the multiple features inherent to DSM, DSMM and GPM (General Purpose Modelling) is solved.**

***Index Terms*—Domain-specific modelling, domain-specific mathematical modelling, general purpose modelling, analytic hierarchy process.**

1. Introduction

The paper proposes Information Technology (IT) for Domain-Specific Mathematical Modelling (DSMM). We can prove the effectiveness of a new IT and corresponding software tools only by their comparison with existing approaches. The paper compares DSMM with IT for General Purpose Modelling (GPM), which prominent example is the use of Unified Modelling Language (UML), as well as with technology of Domain-Specific Modelling (DSM). Analyses shows the reducing number of operations and, accordingly, of the time, needed for metamodels development in the case of application of IT DSMM. The second part of the paper solves the optimization problem of choosing the most effective IT by allocating and comparing the set of features inherited to the technologies of DSM, DSMM and GPM.

The essence of domain-specific modelling is that defined at the metamodel level concepts and operations are semantically match a considered domain. This makes the difference of the domain-specific from the general purpose (unified) languages such as UML [1], EXPRESS-G [2], Merise [3], IDEF0, SADT notations [4] etc., which objects correspond to the concepts of high-level programming languages (in the case of UML – of the object-oriented ones).

This means that the number of operations (and thus, the time spent) for development of models by GPM corresponds to the number of operations used for direct coding. Thus, using "universal" approach does not result in more effective software development, comparatively with traditional way of programming.

On the contrary, a known expert in DSM Steve Kelly argues that in the case of application of UML the process of software development takes 10-15% longer comparatively with direct coding [5]. This is due to the fact that generated from UML models of software code always need manual correction. The necessity to synchronize models and code is one of the most significant shortcomings of UML. Thus, in the case of GPM application we can say about increasing visibility and structuring a software development process, but it does not lead directly to more effective software development.

Rise of efficiency of software development with DSM can be compared with transition from assembler to high-level programming languages [6]. As each concept of a high-level programming language corresponds to a set of commands in assembler, each concept of a metamodel corresponds to the set of commands of a high-level programming language.

Hence, to evaluate effectiveness of domain-specific technology we may compare the quantity of fixed in a metamodel operations with the number of operations of the corresponding software code, was used to implement a model. In particular, the parameter of efficiency may be an amount of software operations corresponding to one concept of a metamodel.

As we shown in [7], development of an additional level of the metamodelling architecture in DSMM allows us to increase the structural correspondence of a model to the modelling domain. This is possible by development of domain-specific meta-metamodels, while IT DSM has a hardcoded one (often graphs based, see e.g. GOPPRR [8]). Implementation of the proposed metamodelling architecture was considered in [9]. It was shown, that IT DSMM increases the compactness of types and related data structures.

Another advantage of IT DSMM is the possibility of development of problems solving methods [10; 11]. It is impossible in IT DSM, where developed by a user metamodels and models of domain are descriptive, and definition of the methods (first of all, for code generators) requires application of an additional high level programming language - for example in MetaEdit+ it is MERL (MetaEdit+ Reporting Language) [12].

In [13] we proposed a method for metamodels development as integrated logical and algebraic systems. In this case, the meta-metamodel defines an algebraic syntax for expressed as a logical system metamodel. This approach allows us to build new and effective procedures to solve problems on logical models with using algebraic methods. For example, using vector algebra as geometrical syntax for syllogistic logic results in development of the metamodel "vector logic". This approach allowed us to implement the method for automation of reasoning from premises, given in vector form [13].

2. MATHEMATICAL MODEL

Here is a theoretical estimation of the number of operations, needed for a metamodel development. In general, the number of operations we can present as the sum of operations needed to define a metamodel's alphabet *A*, a grammar *G* and methods *M*

$$Q = Q_A + Q_G + Q_M \tag{1}$$

Lets consider the number of operations needed for the metamodel development in the case of application of IT DSMM (Q^{DSMM}) and IT DSM (Q^{DSM}) in more details.

Note, that a number of operations to develop a metamodel depends on the specifics of a domain, i.e. its nature, structure and complexity, methods we need to develop, as also skills of a user. We will consider the case of a metamodel development using DSM and DSMM for a same domain, and made by the same user. Let also assume that a number of operations to define the alphabet and the grammar for a metamodel using IT DSM and IT DSMM is approximately the same for a given domain. That is, the number of operations to develop metamodels is depends only on the number of operations to develop their methods.

The number of operations for the development of a method (e.g., code generators) depends on the alphabet and the grammar of a metamodel (in the case of application of IT DSMM) and the alphabet and the grammar of some high-level programming languages (in the case of IT DSM). But, if in the case of IT DSMM the methods of metamodel are based on the alphabet and grammar of the corresponding meta-metamodel, in the case of DSM application the methods should be developed on the base of some external to the metamodel programming language.

It results in the need for additional operations to link a model of domain, developed within the metamodel, with a programming language. Such the "linking" in DSM is done by using an additional scripting languages (like MERL).

Let's introduce the ratio *E* of the number of operations, we need to develop a metamodel using IT DSM - Q^{DSM} and using IT DSMM - Q^{DSMM}. With the previous assumptions, it equals the number of operations needed to develop methods of the metamodels, respectively Q_M^{DSMM} and Q_M^{DSM}:

$$E = Q^{DSM} / Q^{DSMM} = (Q_M^{DSM} + Q_I^{DSM}) / Q_M^{DSMM} \tag{2}$$

In the case of using external programming language we have additional operations Q_I^{DSM}, caused by the need to develop an interface between a domain model and a programming language. These interfaces implement work with external data sources, access to the values of attributes etc. From the ratio $Q_M^{DSM} + Q_I^{DSM} > Q_M^{DSMM}$ follows the increased efficiency of the metamodel development using IT DSMM comparatively with IT DSM. The number of operations for a metamodel development in DSM is always bigger or equal to the number of operations to build a metamodel with DSMM, i.e. $Q_M^{DSM} > Q_M^{DSMM}$.

Let's evaluate the number of operations, needed for the application of the metamodels for development of models of domain. IT DSMM allow us to increase the effectiveness of metamodels not only by rising the structural conformity of their types, but also by increasing *arity* of relationships and of operations of the metamodel.

For example, in the case of development of a geometrical metamodel [15], applying grammar rules to the domain specific types *Sphere* and *Cube* is more effective, than setting the rules over each point of the geometrical sets of the *Sphere* or the *Cube*. E.g. in the case of application of the binary operations on each point of the geometrical sets, the rule of a metamodel grammar can be

$$\forall e_1 \in Sphere \land \forall e_2 \in Cube = \emptyset \tag{3}$$

The number of operations *Q*, needed for calculation (3) is proportional to the product of the number of elements (points) of the geometric sets *Sphere* and *Cube*

$$Q = (|Sphere||Cube|) \tag{4}$$

Definition of the rule of the grammar as a logical operation «and» between types of a *Sphere* and a *Cube* will have the form:

$$Sphere \land Cube = \emptyset \tag{5}$$

Comparison of the rules (3) and (5) allows us to draw a conclusion about increasing efficiency of a metamodel application due to subject adaptation of its grammar rules. In the case of (5) the Boolean operation "and" performed over the geometric sets as a whole, whereas the rule (3) should check for each geometrical point of a *Sphere* and a *Cube* separately.

We also have reducing number of operations for the definition of a metamodel grammar in DSMM. If specification of the grammar in DSM is based on determining rules in terms of binary relationship between concepts of the metamodel, in DSMM it is possible to specify grammars rules of an arbitrary arity.

In addition, the efficiencies of DSMM is associated with its implementation principles, namely, development of multiusers interfaces for organization of work of distributed teams; inclusion of virtual machines in the structure of the DSMM's tools that allow to a user to change the meta-metamodels and the metamodels without recompilation [9].

Generally speaking, IT DSMM includes features of many ITs for computer modelling. The specifics of these ITs depends on considered domains, as well as on the classes of problems to be solved. But we can select the similar ITs and highlight their invariants properties and functionality (for example, use of general purpose or development of domain specific language, application of a visual or a textual notation, implementation of the conceptual or formal modelling, etc.).

Selection and analysis of these criteria allows us to solve the problem of optimizing the decision making, the purpose of which is to choose the most effective IT and relevant toolkit for computer modelling of domains. In the paper we use the method of Saaty - Analytic Hierarchy Process (AHP) [14] to find the solution of the problem. AHP provides a procedure for decomposition of defining the problem criteria with their following processing by pairwise comparisons, that allows us numerically represent the priorities of criteria and, as result, to find a solution. The solution by AHP method is a process of gradual prioritization of criteria by experts of a domain. This criteria should cover all the important characteristics of a problem.

The first stage of AHP is decomposition of a problem into three level hierarchy, starting with the target (the top of the hierarchy), through the evaluation criteria to the lowest level (the list of alternatives).

The purpose of application of AHP: from the number of similar information technologies for computer modelling to choose an IT, which provides the best assessments according to chosen criteria. The selected alternatives in our approach are the following ITs:

A 1. IT of general purpose modelling that uses some unified language (e.g. UML).

A 2. IT of Domain-Specific Modelling (DSM).

A 3. IT of Domain-Specific Mathematical Modelling (DSMM).

As a result of experts work was chosen the set of criteria, needed for evaluation of the properties of ITs for computer modelling. These criteria were divided into several groups, reflecting various aspects of discussed IT (linguistic, procedural, and functional). This was done to follow the formulated in [14] criterion of coherence of the assertions.

The criteria for the evaluation of examined IT in the linguistic aspect are:

K 1. Possibility to develop and to use a specific for domain modelling language (or subject adaptation of a universal modelling language).

K 2. Ability to combine grammars of languages (universal and domain specific, text and visual notations, etc.).

K 3. Possibility to develop both conceptual and formal models.

K 4. Not only declarative (i.e. description of properties and behaviour of objects of a domain), but also imperative modelling (i.e. definition of methods for a problem solving and a computation scheme).

K 5. Absence of transformation of concepts at the transition between stages of modelling (specification → design, design → code etc.).

The criteria that define the time parameters of efficiency of the considered ITs:

K 1. Speed of a model development.

K 2. Time of a metamodel development (or subject adaptation of a universal modelling language).

K 3. Speed of changing models in response to requests.

K 4. Time of transition between stages of software development (from requirements to architecture, from architecture to implementation, etc.).

K 5. Duration of the stage of learning a modelling language.

The criteria, which describe the functionality of the modelling tools:

K 1. Possibility of specification of the method for solution of a class of problems (at the level of metamodel).

K 2. The existence of libraries of standard programming functions (API), as well as predefined methods for solving domain specific problem.

K 3. The possibility of modelling and organization of domain-specific processes.

K 4. Integration of the various stages of a software system development (requirements elicitation, formulating specifications, architectural modelling, etc.).

K 5. The possibility of validation of the models.

At the second phase of AHP by decision makers the matrixes were constructed to compare the relative importance of criteria for establishing their priorities. Evaluation of criteria during their pairwise comparisons assumed the answer: which one is more important (has a greater impact)? For each subjective judgment (based on the knowledge of experts) a quantitative assessment was given.

Thus, each expert produces three comparison tables of the dimension 5x5 (the method for filling these tables is given in the [14]). Because the element $A_{i=j}$ is relatively indifferent to itself, all the elements of the main diagonal of the matrices A for pairwise comparisons have a value, equal to one.

The total number of pairwise comparisons made for each table is $\frac{n\times(n-1)}{2}$. To reduce the size of the paper we will give only the table of comparisons and calculations of the linguistic aspect of the ITs (see table 1).

Let us calculate the vector of priorities for the matrix of pairwise comparisons. For the estimation of the components of the vector the geometric mean for each row of the matrix A was calculated by the formula

$$b_i = \sqrt[n]{\prod_{k=1}^{n} a_{ik}} \, . \qquad (6)$$

The obtained by the formula (6) column of numbers was normalized by dividing each number b_i on the sum of all elements in the column, as result the values of the components of the vector of local priorities were obtained

$$x_i = b_i / B, \qquad i = \overline{1, n}, \tag{7}$$

where $B = \sum_{i=1}^{n} b_i$.

For all matrices of pairwise comparisons the accuracy of calculations was evaluated by the formula (8)

$$\delta_x = \left| 1 - \sum_{i=1}^{n} x_i \right| \times 100\%. \tag{8}$$

TABLE 1. MATRIX OF PAIRWISE COMPARISONS FOR THE LINGUISTIC ASPECT OF ITS

Criterion		K1	K2	K3	K4	K5	Geom. mean	Weight
Possibility to develop and to use a specific for domain modelling language	K1	1	3	7	5	5	2,18	0,37
Possibility to combine grammars of languages	K2	1/3	1	7	5	3	1,55	0,26
Possibility of development of conceptual and formal models	K3	1/7	1/7	1	1/3	1/3	0,46	0,08
Possibility of declarative and imperative modelling.	K4	1/5	1/5	3	1	3	0,88	0,15
Absence of transformations of concepts at the transition between stages of modelling.	K5	1/5	1/3	3	1/3	1	0,71	0,12
Sum							5,78	0,98
Calculation error								0,02

Let's analyse the vectors of priorities for the matrix of pairwise comparisons at the second level.

The values, obtained for components x_i of the vector of local priorities allow us to rank criteria according to the estimations of experts in the descending by derived weights order.

1. Ability to develop and to use a specific for domain modelling language (weight - 0,3767).
2. Possibility to combine grammars of the modelling languages (weight - 0,2685).
3. Possibility of declarative and imperative modelling (weight - 0,1515).
4. Absence of transformations of concepts at transition between stages of the modelling (weight - 0,1227)
5. Possibility of development of conceptual and formal models (weight - 0,0804).

Note that a value of the criterion, which received the lowest score («Possibility of development of conceptual and formal models»), cannot be neglected, because its weight is 8% of the total weight of all the criteria.

Let us develop a matrix of pairwise comparisons for the third level. For each criterion were conducted pairwise comparison of alternatives and in accordance with the formula (6), (7) the synthesis of local priorities was made. Table 2 shows the pairwise comparison of alternatives by the criterion K1 «Possibility to develop and to use a specific for domain modelling language».

TABLE 2. MATRIX OF PAIRWISE COMPARISONS BY CRITERION "THE POSSIBILITY TO DEVELOP AND TO USE A SPECIFIC FOR DOMAIN MODELLING LANGUAGE"

Alternative		A1	A2	A3	Geom. mean	Weight (mean /sum)
IT of General Purpose Modelling (e.g. UML).	A1	1	1/7	1/9	0,59	0,18
IT Domain-Specific Modelling (DSM).	A2	7	1	1	1,27	0,40
IT Domain-Specific Mathematical Modelling (DSMM).	A3	9	1	1	1,31	0,41
Sum					3,17	0,99
Calculation error						0,01

Based on the conducted calculations, the A3 alternative - IT of Domain-Specific Mathematical Modelling took the first place (with weight 0,4129) by the criterion "Ability to develop and to use a specific for domain modelling language". Similar calculations were also done for other criteria.

Let us make a synthesis of global priorities of ITs for modelling. Table 3 shows the source data for the calculation of the values of the components of the vector of global priorities that were received on the previous phases of AHP.

For calculation of global priorities of ITs in the table 2 the local priorities of alternatives were located according to each criterion; each column of vectors of alternatives was multiplied by the priority of the corresponding criterion, and results have been added along each row:

$$V^{M_j} = \sum_{i=1}^{n} x_i \times z_i^{M_j}. \tag{9}$$

The results of the calculations by the formula (9) can be interpreted as a utility function for each of the alternatives. The first place has A3 IT Domain-Specific Mathematical Modelling, that outperformed the nearest neighbour «A3 IT Domain-Specific Modelling» on (0,4726-0,3410)× 100%=13,16% (in the linguistic aspect of consideration).

TABLE 3. COMPUTATION OF GLOBAL PRIORITIES (LINGUISTIC ASPECT)

Criterion		K1	K2	K3	K4	K5	Global priority
Alternative		0,38	0,26	0,08	0,15	0,12	
GPM	A1	0,18	0,23	0,18	0,18	0,18	0,19
DSM	A2	0,40	0,23	0,34	0,31	0,41	0,34
DSMM	A3	0,41	0,54	0,47	0,54	0,41	0,47
Sum							1,0
Calculation error							0,00

Figure 1 shows the chart comparing ITs for modelling domains in the linguistic aspect.

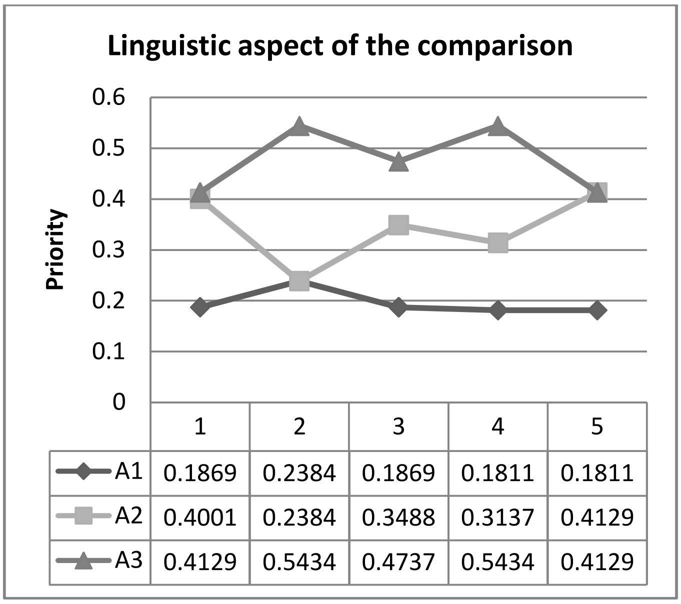

Fig.1. Chart comparing linguistic aspect of the ITs

CONCLUSIONS

The evaluation of efficiency of Information Technology for Domain-Specific Mathematical Modelling is given. IT DSMM was compared with technologies of general purpose modelling and domain specific modelling.

The increasing efficiency of software development using IT DSMM is caused by the fact that each concept of its metamodel corresponds to the set of commands of a high-level programming language.

It is proven that the number of operations for a metamodel development with DSM is always bigger or equal to the number of operations to build a metamodel using IT DSMM. This results from the fact that DSM applies an external programming language and therefore has additional operations, needed for development of an interface between a metamodel and a programming language.

The similar to DSMM technologies of computer modelling were allocated and criteria for comparing them were defined. These criteria were divided into several groups that reflect the linguistic, the procedural and the functional aspects of analyses. By the AHP method the optimization problem of decision making is solved, what allows us to choose the most efficient information technology. The best evaluation according to the expert's ratings has IT DSMM (comparatively with DSM and general purpose modelling).

Prospects for further research. In our following works we will address the methods for development of metamodels as logical and algebraic systems, as the continuation of the results obtained in [13].

REFERENCES

[1]. Grady Booch, James Rumbaugh, Ivar Jacobson. The Unified Modeling Language User Guide (2nd Edition), Addison-Wesley Professional. 2005. 496 p

[2]. Douglas A. Schenck and Peter R. Wilson. Information Modelling the EXPRESS Way. Oxford Univ. Press, 1993. 416 p.

[3]. Avison D. MERISE: A European Methodology for Developing Information Systems. European Journal of Information Systems. Jan. 1991, p. 183-191.

[4]. David A. Marca, Clement L. McGowan. IDEF0 and SADT: A Modeler's Guide. OpenProcess, Inc. -2005. - 392 p.

[5]. http://www.metacase.com/blogs/stevek/blogView

[6]. DSM forum. - http://www.dsmforum.org/why.html

[7]. Vitaliy Mezhuyev. Metamodelling Architecture for Modelling Domains with Different Mathematical Structure. Advanced Computer and Communication Engineering Technology. Lecture Notes in Electrical Eng., 2015. Vol. 315. Pp. 1049-1055.

[8]. Steven Kelly and Juha-Pekka Tolvanen. Domain-Specific Modelling: Enabling Full Code Generation. Wiley-IEEE Computer Society Pr. 2008. 427 p.

[9]. Vitaliy Mezhuyev. Architecture of software tools for Domain-Specific Mathematical Modelling. Proceedings of 2014 International Conference on Computer, Communication, and Control Technology (Malaysia, Sept 2-4, 2014). Pp. 166-170.

[10]. Vitaliy Mezhuyev. Methodology of Domain Specific Mathematical Modelling. Proceedings of the 3rd International Congress on Natural Sciences and Engineering ICNSE 2014 (Kyoto, Japan, May 7-9, 2014). - Pp. 54-64.

[11]. Vitaliy Mezhuyev. Theory and Applications of Domain Specific Mathematical Modelling. Proceedings of the 3rd International Conference on Computer Engineering & Mathematical Sciences ICCEMS 2014 (Langkawi, Malaysia, 4-5 December 2014). Pp. 853-874.

[12]. MERL - MetaEdit+ Reporting Language. - http://www.metacase.com/support/45/manuals/mwb/Mw-5_3.html

[13]. Vitaliy Mezhuyev. Development of Metamodels as Logical and Algebraic Systems. Proceedings of the 2014 International Conference on Information Science, Electronics and Electrical Engineering ISEEE (Hokkaido, Japan, April 26-28, 2014). – pp. 1850-1855.

[14]. Thomas L. Saaty. Fundamentals of Decision Making and Priority Theory with the Analytic Hierarchy Process (Analytic Hierarchy Process Series, Vol. 6). RWS Publ.; 2000. 477 p.

[15]. Vitaliy Mezhuyev, Refik Samet. Geometrical Meta-metamodel for Cyber-Physical Modelling. Proceedings of International Conference Cyberworlds 2013 (Yokohama, Japan, October 21-23, 2013). - pp. 89-93.